**Desafío I: AUTOMATIZACIÓN DE PRUEBAS A SERVICIOS WEB RESTFUL - BOOKER**

1. Para la identificación de casos de prueba se elaboró una tabla decisiones en las cuales se consideraron escenarios Happypath y UnHappypath. A continuación, se detalla el plan de pruebas realizado:

URL del repositorio: https://github.com/CandyEsp/RestFull.git

**Plan de Pruebas - RestFull Booker**

**Visión General**

RestFull Booker es una aplicación que permite crear, leer, actualizar y eliminar reservas. Actualmente se están detectando diferentes errores funcionales a nivel de la GUI por lo que se requiere que se implementen pruebas automatizadas a los servicios web.

**Alcance**

El presente plan de pruebas contemplará pruebas manuales y automatizadas a nivel de servicios.

**Riesgos Identificados**

|  |  |  |  |
| --- | --- | --- | --- |
| Funcionalidad | Riesgo | Número de prioridad | Estrategia |
| Get BookingIds | Filtrar por parámetro incorrecto | 3 | Manual |
| Get BookingIds | Filtrar por parámetro correcto | 2 | Manual |
| Get Booking by ID | Filtrar por ID Existente | 1 | Automatizada |
| Get Booking by ID | Filtrar por ID No Existente | 3 | Manual |
| Create Booking | Crear reserva con parámetros correctos en el Header, requestBody y objeto no obligatorio | 1 | Automatizada |
| Create Booking | Crear reserva con parámetros correctos en el Header, requestBody sin colocar el objeto no obligatorio | 1 | Automatizada |
| Create Booking | Crear reserva sin colocar los datos obligatorios en el request body | 1 | Automatizada |
| Create Booking | Crear reserva sin colocar los datos obligatorios en el Header | 3 | Manual |
| Update Booking | Actualizar reserva con campos correctos en el Header, envío de token y valores obligatorios en el request body | 1 | Automatizada |
| Update Booking | Actualizar reserva con campos correctos en el Header, envío de token y sin colocar valores obligatorios en el request body | 3 | Manual |
| Update Booking | Actualizar reserva con campos correctos en el Header, sin envío de token y colocando valores obligatorios en el request body | 2 | Automatizada |
| Update Booking | Actualizar reserva con campos incorrectos en el Header, envío correcto de token y colocando valores obligatorios en el request body | 3 | Manual |
| Update Booking | Actualizar reserva con campos incorrectos en el Header, sin envío de token y colocando valores obligatorios en el request body | 3 | Manual |
| Partial Update Booking | Actualizar campos de reserva enviando el Header correcto, token e indicando los valores a actualizar | 1 | Automatizada |
| Partial Update Booking | Actualizar campos de reserva enviando el Header correcto, sin enviar el token e indicando los valores a actualizar | 1 | Automatizada |
| Partial Update Booking | Actualizar campos de reserva enviando el Header incorrecto, enviando token correcto e indicando los valores a actualizar | 3 | Manual |
| Delete Booking | Eliminar ID de reserva existente y sin el envío de token. | 3 | Manual |
| Delete Booking | Eliminar ID de reserva no existente y con el token correcto | 1 | Automatizada |
| E2E | Se requiere realizar una prueba E2E en la cual se consulte el ping de la url, se Cree una reserva, se actualice y posteriormente se Elimine | 1 | Automatizada |

\*Para la identificación de riesgos se utilizó la técnica “Tabla de decisiones” para evaluar las funcionalidades del aplicativo.

**Módulos a probar**

A continuación, se indican las funcionalidades consideradas dentro del plan de pruebas:

* Get BookingIds
* Get Booking by ID
* Create Booking
* Update Booking
* Partial Update Booking
* Delete Booking

1. Para la codificación del script automatizado se utilizó como lenguaje Java, como compilador Maven y las librerías de Junit y Cucumber. Para la visualización del reporte se utilizó Serenity integrado con funcionalidades de RestAssured.
2. Se identificaron 19 casos de prueba de los cuales se automatizaron 10 brindando una cobertura del 53%.
3. Cada escenario que se puede ejecutar está identificado por un tag, a continuación, se nombran los siguientes:

* @CreateBookingCorrect: Crear una reserva correcta
* @CreateBookingInCorrect: Crea una reserva ingresando datos incorrectos en el body
* @DeleteBookingIncorrecto: Elimina un ID de reserva no existente
* @EndToEnd: Realiza el ping, crear, obtener y eliminar una reserva
* @GetBookingByID: Obtiene una reserva por ID
* @GetBookingID: Obtiene el listado de reservas
* @PartialUpdateCorrect: Se realiza el update parcial correcto de la reserva
* @PartialUpdateInCorrect: Se realiza la petición sin enviar el token
* @UpdateBookingCorrect: Realiza la petición que permite actualizar una reserva
* @UpdateBookingSinToken: Realiza la petición sin enviar el token

1. Para la realización del script se está utilizando el patrón de diseño POM. En el paquete main está la clase BaseClass donde se encuentran métodos reutilizables para generar las peticiones. En el paquete test se encuentran alojados los Steps, el Runner y los features. Se tiene una carpeta Schemas donde se encuentran archivos json que se utilizan para realizar y validar las peticiones realizadas.

Interfaz de usuario gráfica, Texto, Chat o mensaje de texto

Descripción generada automáticamente

1. El reporte en serenity se visualiza de la siguiente manera:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. También se incluyó en el proyecto el archivo jenkinsfile el cual permite ejecutar el proyecto de acuerdo al escenario especificado dentro de la configuración.

Texto

Descripción generada automáticamente

1. Para fines de revisión para este ejercicio se entrega:

* La colección de postman en json **Restfull.postman\_collection.json**
* Tabla de decisiones en Excel **RestFullApi.xlsx**
* Archivo comprimido del script **framework-api.zip** (También se puede descargar del repositorio de git <https://github.com/CandyEsp/RestFull> )